Technische Universiteit
Eindhoven
University of Technology

Department of Mechanical Engineering
Control System Technology Research Group

Implementation and comparison of linearization-based and backstepping controllers for quadcopters

Master Thesis

Jesús Valle Rodríguez

Supervisor:
D. J. Guerreiro Tomé Antunues

Eindhoven, April 2017

Abstract

In this work, two approaches to the control of a quadcopter are followed. The first approach resorts to linear quadratic control (LQR) techniques and is based on the linearization of the quadcopter dynamics. Motivated by the fact that this linearization results in decoupled dynamics for the longitudinal, lateral, height and yaw axis, the LQR controllers can be designed separately. Moreover, the controllers for the longitudinal and lateral dynamics exploit the cascaded structure of the model. The second approach resorts to non-linear control and exploits the fact that the full non-linear model of the quadcopter also has a cascaded structure: the torque inputs control the angles which in turn determine the forces which drive the position states. The approach is based on a widely used non-linear control design technique for cascaded systems known as back-stepping. Simulations of the two approaches are carried out and conclusions are drawn on the pros and cons of each approach.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Motivation

In recent years the use of Unmaned Aerial Vehicles (UAVs), or simply drones, has become more popular. What started as a technology mostly available and developed by the military it has now become a reality in the civil industry. UAVs are starting to be implemented in a multitude of different scenarios and expected to play a prominent role in some industries where its characteristics suppose a great improvement compared to technology or methods used today.

In fact, one of the most common applications of aerial robots is chemical crop spraying, which was promoted by Japan companies in the 1980's [1, 3]. Since then payload delivering has become a major area in which drones have become more popular: from delivering punctual products in areas hard to reach for humans to transportation of small and large cargo, firefighting and disaster response.

The fact that drones are being used for such activities more often opens the possibility to other civil and private applications. The strong air-space regulations still do not contemplate the spread use of drones. At first only the long-term scientific projects could obtain special permits. Still, the wide-spreading of surveillance drones applications for low and medium-high altitude and remote monitoring have pushed some changes in the air-space regulations [19].

Nowadays several companies aim to take profit of this technology and apply it directly to their business-core: logistic companies that want to deliver their packages with drones [18] and other companies that aim to turn quadcopters in a feasible way of personal/private transport in the years to come [17].

Nowadays, several types of UAVs are being developed. Yet, it is possible to state that the two main groups of drones are airplane-drones and multirotors. While the first type presents similar characteristics to the airplane structure, which tries to replicate at a different scale, the second type has opened a new interesting research field which has attracted a lot of attention in recent years.

Similar to UAVs in terms of mechanics and dynamics are the helicopters. However, the most common multirotors that we can find today differ from conventional helicopters. In fact, for quadcopters, the blades have a constant pitch and have no tail rotor, which generates no useful thrust. The rotors of a multirotor counteract the effect of counter-rotation by spinning two opposite rotors at the same speed, one clockwise and the other counter-clockwise. This principle leads to a disposition of a pair number of rotors. The most common multirotors are the quadcopters or quadrotors, multirotors made of four rotors in the same plane.

The prospects for drones, and quadcopter in particular, are very high as they represent a new and more affordable way of conquering the air. We are at the beginning of a new technology which means that the work, research and development carried out today on this area will highly determine how we think of the not-so-far future of these UAVs.

Goal

The goal of this project is to test and compare different strategies to control a quadcopter. The project tackles the modelling of the quadcopter dynamics and the different control approaches. The control structures derived aim to control the quadcopter along a pre-computed path.

Different control strategies will be derived and simulated with the dynamics of a quadcopter. Each control strategy will be developed using a different approach in order to test its suitability. Moreover, each control strategy will imply different implementations on the real system. All the results of the simulations and its implications will be commented along the thesis.

Main Findings

In this thesis the nonlinear dynamics of a quadcopter will be modelled. The inherent instability of the quadcopter calls for a control strategy to stabilise it. Three control approaches will be developed that not only guaranties stability but allows the quadcopter to track a given path. The first two of these control structures will be linear while the third will take into account the highly nonlinear dynamics of the quadcopter. All control strategies are discussed and compared, not only for the results obtained but also for the implications they have when developing the control structure.

The system used as model to be studied is the commercial quadcopter Parrot AR Drone 2.0. All mathematical calculations and simulations are carried out in *Matlab*.

As shown throughout the thesis the different control strategies were successfully derived and proven to work in a simulation environment. First the nonlinear dynamics are modelled by the Newton-Euler equations. The resulting non-linear model explicitly shows the high coupling between state variables as well as its cascade structure. Each of the different control strategies, linear and nonlinear, derived for controlling the quadcopter model are proved to control the system along the tracking problem.

Organisation of the thesis

The outline of the thesis is:

- **Chapter 2. Modelling of a Quadcopter**

  In this chapter the nonlinear dynamics are modelled. The resulting equations are studied to show the coupling of position and attitude variables and the implications it has when controlling its unstable dynamics.

- **Chapter 3. Control of a Quadcopter**

  Different approaches to the control of the quadcopter are studied in this chapter. First, two linear controllers are derived and compared, and later a nonlinear controller is proposed based on backstepping techniques for building a Lyanpunov function able to guarantee stabilization of the nonlinear dynamics.

- **Chapter 4. Simulations**

  The proposed controllers in Chapter 3 are simulated to successfully track a given trajectory in space. The parameters related to the Parrot AR Drone 2.0 are introduced in the model for a more accurate and realistic system response as well as the physical limitations of this commercial drone. Smooth trajectories are used for simulating the response of the quadcopter dynamics due to the modelled nonlinearities.

Chapter 2

Modelling of a Quadcopter

Quadrotors are multirotor helicopters built with 4 rotors on the same plane. These mechanically simple flying robots are able to take-off and land vertically as well as being able to perform very agile trajectories and hover in the air. Due to its mechanical design of four rotors in a same plane the drone is under-actuated to move in the 6D configuration space (x, y, z, and orientation *roll*, *pitch* and *yaw*). Moreover, the design of the quadcopter makes it an unstable system. This allows it to perform complex trajectories and manoeuvres in the air, but also calls for the need of a feedback controller to stabilize the system and avoid that it crashes. In order to control and stabilize the quadcopter some approaches are done with a linear controller design around an equilibrium point or hover. Nowadays however, a lot of research is being carried out implementing different techniques of non-linear control strategies [21, 22, 15, 10, 6, 12].

In this chapter the model and dynamics of the quadcopter will be discussed. The control approaches considered later, linear and non-linear, will be derived from the mathematical model of the quadcopter developed in this chapter.

2.1 System identification

For this project an 'X-type' quadcopter will be used. In Figure (2.1) the symmetrical configuration and its four rotors are shown. Rotors are attached to the body frame at a distance l from the centre of mass of the quadcopter. The direction of rotation, clockwise and counter-clockwise, of each rotor, shown in Figure (2.1), is equal for opposite rotors. The angular velocity of each rotor can be controlled individually. This allows to control the resulting thrust and torques generated in each direction.

Let $\{A\}$ denote the universal reference, or inertial frame, with $\{e_x^A, e_y^A, e_z^A\}$ as unitary vectors defining the reference coordinates. The position of the centre of mass of the quadcopter can be defined as $\xi = (x, y, z)^T$ with respect to the inertial frame $\{A\}$.

Let $\{B\}$ denote the body fixed reference with $\{e_x^B, e_y^B, e_z^B\}$ as unitary vectors attached to the centre of mass of the quadcopter. The orientation of the body frame can be transformed into universal frame using the Euler angle parametrization. Therefore, the orientation of the quadcopter, or attitude, is defined by Euler angles in the inertial frame as $\lambda = (\phi, \theta, \psi)^T$, where ϕ, θ, $\psi \in \Re$ stand for *roll*, *pitch* and *yaw* angles respectively.

Let $\Omega = (p, q, r)^T$ define the rotational velocity of the quadcopter with respect to the body frame $\{B\}$.

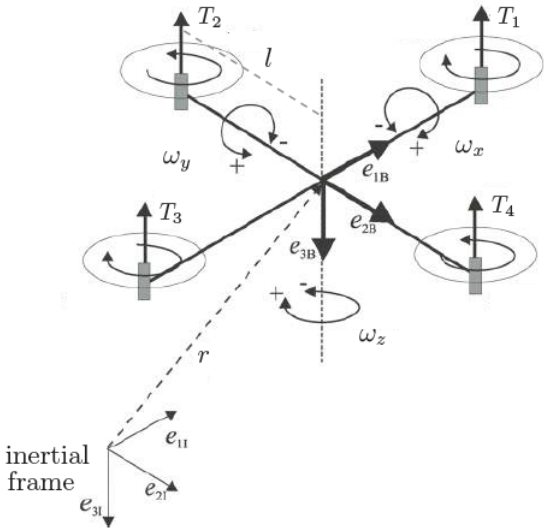


Figure 2.1: Configuration of the quadrotor: inertial and body frame, quadrotor position, rotors thrusts and angular torques.

A rotation matrix R, equation (2.1), composed of rotations around each body frame axes, transforms the orientation of the body frame $\{B\}$ to the inertial reference frame $\{A\}$. Using the Z-Y-Z Tait-Bryan angles, the following rotation matrix is obtained.

$$\begin{aligned} R(\lambda) &= R_z(\psi)R_y(\theta)R_x(\psi) \\ &= \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\theta\sin\psi & \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi \\ \cos\theta\sin\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & \cos\phi\sin\theta\sin\psi - \cos\psi\sin\phi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \end{aligned} \tag{2.1}$$

For transforming rotational velocities between the two frame another transformation is required

$$\dot{\lambda} = Q(\lambda)\Omega \tag{2.2}$$

where the matrix $Q(\lambda)$ is derived in [14].

2.2 Dynamic model

For model simplicity, it is assumed that the rotor propellers are rigid, as well as the quadcopter body frame. Aerodynamic effects as blade flapping and air flow disruptions induced by ground effect will be neglected.

The nonlinear model of the quadcopter can be described by the Newton-Euler equations. These can be seen as a position subsystem (2.3a) and an attitude subsystem (2.3b):

$$\begin{cases} \dot{r} = v \\ m\dot{v} = mge_3^A - R(\lambda)Te_3^B \end{cases} \tag{2.3a}$$

$$\begin{cases} \dot{\lambda} = Q(\lambda)\omega \\ J\dot{\omega} = \tau - \omega \times J\omega \end{cases} \tag{2.3b}$$

where $m \in \mathbb{R}$ is the mass of the quadcopter, $g \in \mathbb{R}$ the gravitational constant, and $I \in \mathbb{R}^{3x3}$ the inertia matrix of the quadcopter in the body frame $\{B\}$. Due to the symmetry of the quadcopter the inertia matrix can be assumed to be diagonal.

T is the sum of the thrusts of each rotor along the z axis of the body frame and $\tau = [\tau_x, \tau_y, \tau_z]^T$ are the torques applied to the quadcopter by the rotors.

Assuming that the thrust generated by each rotor is proportional to the square of its rotor speed, where Ω_i $i = 1...4$ stand for each rotor of the quadcopter. The relation between the each rotor speeds and the Thrust and the Torques considered as inputs at (2.3) can be expressed as

$$\begin{bmatrix} T \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \underbrace{\begin{bmatrix} c_T & c_T & c_T & c_T \\ 0 & lc_T & 0 & -lc_T \\ lc_T & 0 & -lc_T & 0 \\ c_Q & -c_Q & c_Q & -c_Q \end{bmatrix}}_{\Gamma} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \tag{2.4}$$

where c_T stands for thrust coefficient, c_Q for drag coefficient and l in the distance from the propellers to the rotation axis. Both coefficients are positive and depend on the the rotors and blades geometry and air properties, and are determined experimentally.

It can be shown that the matrix Γ is invertible. By inverting it, the individual rotor speeds can be computed from the desired model inputs.

2.3 Cascade system structure

Observing 2.3 it can be seen that the position subsystem depends on the attitude subsystem. However the attitude subsystem only depends on its own variables. This fact allows to implement a cascade control strategy [6] as it will be shown in Chapter 3.

In Chapter 3 different controllers will be derived to control the model of the quadcopter. As it will be shown, the designed controllers will take advantage in different ways of this cascade system structure.

For all controllers derived, the control problem for this subsystem will always refer to the control the position, ξ, and *yaw*, ψ, of the quadcopter. The remaining variables, ϕ and θ, which partially define the attitude of the controller, are expected to remain stable and as a consequence, stabilise the *xy*-plane of the quadcopter.

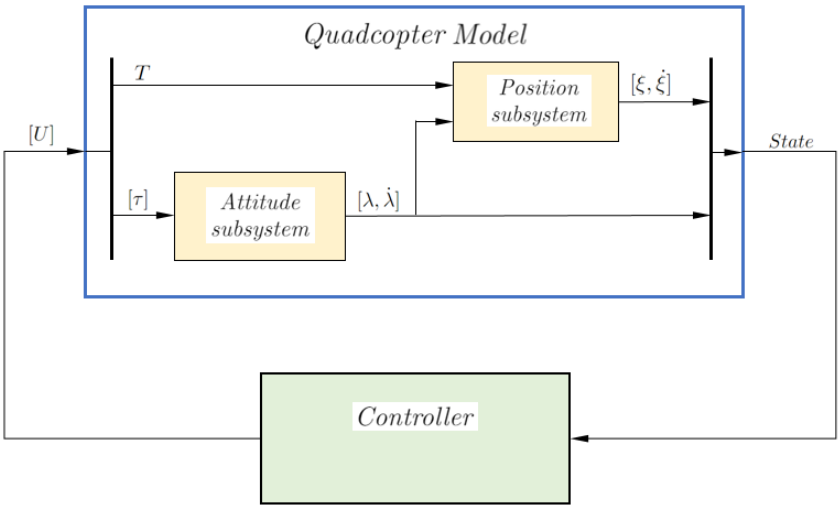


Figure 2.2: Cascade structure of the quadcopter model.

Chapter 3

Control of a Quadcopter

In this chapter three different approaches for designing a linear controller will be discussed. First, two linear controllers will be derived using the Linear Quadratic Regulator (LQR) framework. Also, as it will be seen, for both cases the controller will result in a PD (Proportional-Derivative) structure. Later, a nonlinear control strategy based on a Lyapunov function will be proposed. The quadcopter models derived in this chapter are only used for deriving the controllers. The testing of the control strategies on the non linear model, equation (2.3) are carried out in Chapter 2.2, where the parameter of the controllers are also defined, if necessary.

For the first proposed linear controller a cascade controller is computed. This controller structure results of the approach taken when analysing the quadcopter model. The inner loop aims to stabilise the attitude response of the quadcopter, while the outer loop aims to control the quadcopter to a desired position in space.

The second proposed linear controller does not make use of the cascade structure of the quadcopter model but takes into account the coupling between both position and attitude subsystem. The coupling of this subsystem derives in the design of a single feedback loop but with 4 PD controllers in parallel: each one for the 4 inputs that the model has.

For all cases in this chapter the control problem aims to track a trajectory in space. The trajectory is defined in the 4D subspace:

$$\mathscr{T} := \left(\xi(t)^d, \psi(t)^d\right) \in R^4 \tag{3.1}$$

The remaining attitude variables, ϕ and θ, of the model are implicitally determined.

3.1 Cascade control

As mentioned in the previous Chapter 2, the model allows to implement a cascade control structure without many variable transformations. For this case the controller design is approached in two phases: stabilise the attitude variables in an inner loop, and track a given trajectory in space in the outer loop.

With this approach, the control problem is divided into a tracking problem along a position in a 4D subspace and regulation problem for a 2D attitude subspace.

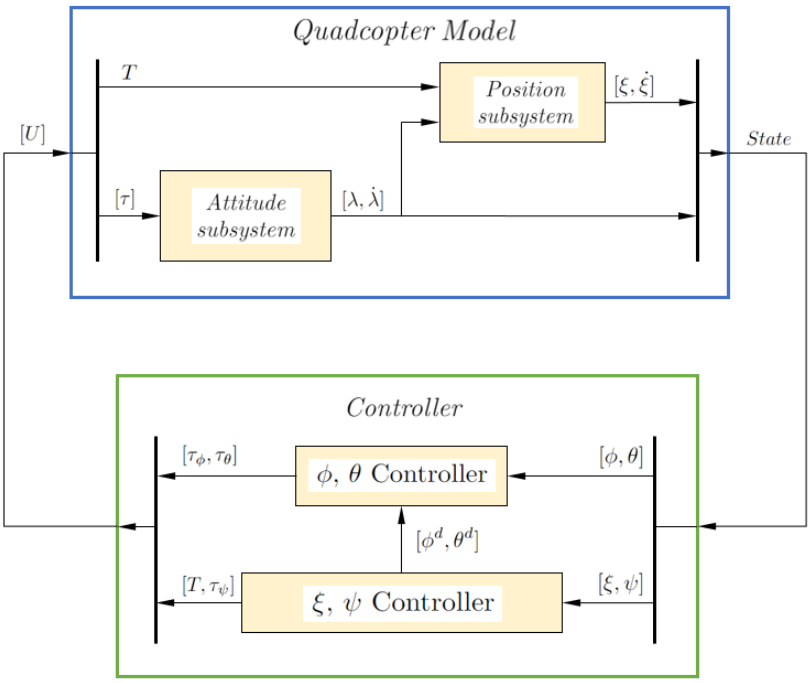


Figure 3.1: Quadcopter model with a cascade controller.

Outer loop

The outer loop of the controller aims to control the tracking problem of the system along a given trajectory. The trajectory is given in a 4D subspace encompassing the *position* and *yaw* of the quadcopter.

$$\mathscr{T} := \left(\xi(t)^d, \psi(t)^d\right) \in R^4$$

The equations for the outer loop for designing the controller are:

$$\begin{pmatrix} m\ddot{x} \\ m\ddot{y} \\ m\ddot{z} \\ I_z\ddot{\psi} \end{pmatrix} = \begin{pmatrix} T_x \\ T_y \\ T_z \\ \tau_z \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \\ mg \\ 0 \end{pmatrix} \tag{3.2}$$

where T_x, T_y and T_z represent the Thrust component in each direction in the inertial frame $\{A\}$.

Note that the cascade control structure implies that the variables ϕ and θ, responsible for the regulation of the attitude do not appear explicitly when modelling the outer loop system. Thus, the subsystems appear to be decoupled.

With this model, a *LQR* controller is computed. As we will see, the obtained controller structure results to be a PD controller.

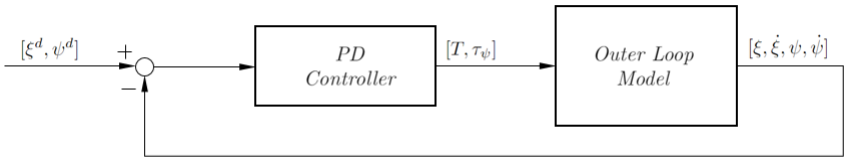


Figure 3.2: Outer loop control diagram.

Inner loop

As mentioned before, the position and attitude subsystems seem to be decoupled. However, to be able to generate thrust in any direction different from the z-axis, the xy-plane of the quadcopter must tilt, i.e. the attitude subsystem controls the way the thrust controls the position of the quadcopter, the subsystems are coupled implicitly.

In order to know how much the quadcopter has to tilt, the relation between the Thrust components and the change of base between the inertial and the body frame is exploited:

$$\begin{pmatrix} T_x \\ T_y \\ T_z \end{pmatrix}_A = R(\lambda) \begin{pmatrix} 0 \\ 0 \\ T \end{pmatrix}_B \tag{3.3}$$

Form this equation (3.3), and considering T_x, T_y and T_z as known, the desired ϕ^d and θ^d for each instant can be computed.

The desired attitude variables turn out to be the reference for the inner loop controlled system. Analogous to the outer loop, the inner loop linearised system used design the controller is:

$$\begin{pmatrix} I_x \ddot{\phi} \\ I_y \ddot{\theta} \end{pmatrix} = \begin{pmatrix} \tau_x \\ \tau_y \end{pmatrix} \tag{3.4}$$

With this model, a LQR controller is computed. The obtained controller structure results to be a PD controller.

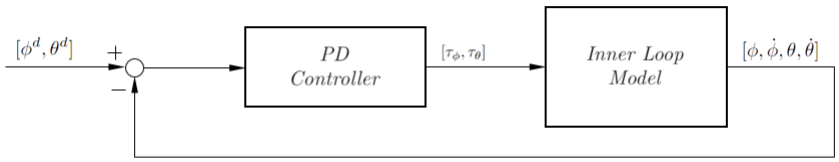


Figure 3.3: Inner loop control diagram.

Cascade control

For integrating both designed controllers a conversion of variables has to be carried out. In order to do that, the desired ϕ^d and θ^d must be derived from equations (2.3a). By linearizing the rotational matrix, equation (2.1), and substituting $\phi = \phi^d$ and $\theta = \theta^d$ a linear transformation between the thrust and its components in the inertial frame is obtained

$$\begin{aligned} T &= T_z + mg \\ \phi^d &= \frac{-T_y}{T} \qquad \theta^d = \frac{T_x}{T} \end{aligned} \tag{3.5}$$

Taking into account these relations, the control law for the outer loop can be expressed as

$$T_z = \begin{bmatrix} kP_z & kD_z \end{bmatrix} \begin{bmatrix} z_r - z \\ \dot{z}_r - \dot{z} \end{bmatrix} \tag{3.6}$$

$$T_x = \begin{bmatrix} kP_x & kD_x \end{bmatrix} \begin{bmatrix} x_r - x \\ \dot{x}_r - \dot{x} \end{bmatrix} \tag{3.7}$$

$$T_y = \begin{bmatrix} kP_y & kD_y \end{bmatrix} \begin{bmatrix} y_r - y \\ \dot{y}_r - \dot{y} \end{bmatrix} \tag{3.8}$$

$$\tau_z = \begin{bmatrix} kP_\psi & kD_\psi \end{bmatrix} \begin{bmatrix} \psi_r - \psi \\ \dot{\psi}_r - \dot{\psi} \end{bmatrix} \tag{3.9}$$

and the control law for the inner loop as

$$\tau_x = \begin{bmatrix} kP_\phi & kD_\phi \end{bmatrix} \begin{bmatrix} -\frac{T_y}{T} - \phi \\ -\dot{\phi} \end{bmatrix} \tag{3.10}$$

$$\tau_y = \begin{bmatrix} kP_\theta & kD_\theta \end{bmatrix} \begin{bmatrix} \frac{T_x}{T} - \theta \\ -\dot{\theta} \end{bmatrix} \tag{3.11}$$

By combining equations (3.6) to (3.11) in a cascade structure, the control input $U = [T, \tau_x, \tau_y, \tau_z]$ is obtained.

Knowing these conversions, the integration of the inner and outer loop result in:

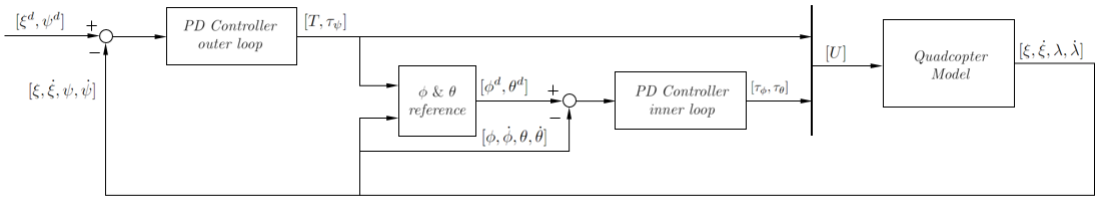


Figure 3.4: Cascade controller with ϕ and θ explicit computation.

In order for this control structure to be successful the inner loop has to be converge faster to its desired input so that any unwanted transient in the control of the attitude subsystem does not unstabilises the quadcopter.

3.1.1 Model linearization control

For this second approach for designing a linear controller the cascade structure of the system model is not considered and instead the full model (2.3) is linearised around an equilibrium point. The linearization of the model is carried out in order to obtain a linear model with the following structure:

$$\begin{cases} \delta\dot{x} = A\delta x + B\delta u \\ \delta y = C\delta x \end{cases} \tag{3.12}$$

where

$$\{(\bar{x},\bar{u}) \mid f(\bar{x},\bar{u}) = 0\} \rightarrow (\bar{x},\bar{u}) \text{ is an equilibrium point}$$

$$\begin{aligned}
\delta x &= x - \bar{x} \\
\delta u &= u - \bar{u} \\
A &= \left.\frac{\partial f(x,u)}{\partial x}\right|_{x=\bar{x},u=\bar{u}} \\
B &= \left.\frac{\partial f(x,u)}{\partial u}\right|_{x=\bar{x},u=\bar{u}} \\
C &= \left.\frac{\partial g(x,u)}{\partial x}\right|_{x=\bar{x},u=\bar{u}}
\end{aligned}$$

The linear equations obtained are:

$$\begin{aligned}
\dot{\xi} &= v \\
m\dot{v} &= \begin{bmatrix} -mg\theta \\ mg\phi \\ -(T-mg) \end{bmatrix} \\
\dot{\lambda} &= \Omega \\
I\dot{\Omega} &= \tau
\end{aligned} \tag{3.14}$$

These equations can be decoupled into 4 subsystems. Attitude and position variables can be found coupled in the same subsystem. Each subsystem has as an input one of the inputs of the quadcopter model, equation (2.3), and therefore, each subsystem defines the response of the quadcopter around a hover point for each input. The decoupled linearised equations are:

i. *Pitch* control:

$$\begin{cases} \dot{x} = V_x \\ \dot{V}_x = -mg\theta \\ \dot{\theta} = q \\ \dot{q} = \dfrac{\tau_x}{I_x} \end{cases} \tag{3.15}$$

ii. *Roll* control:

$$\begin{cases} \dot{y} = V_y \\ \dot{V}_y = mg\phi \\ \dot{\phi} = p \\ \dot{p} = \dfrac{\tau_y}{I_y} \end{cases} \tag{3.16}$$

iii. Altitude control:

$$\begin{cases} \dot{z} = V_z \\ \dot{V}_z = -\dfrac{T-mg}{m} \end{cases} \tag{3.17}$$

iv. *Yaw* control:

$$\begin{cases} \dot{\psi} = r \\ \dot{r} = \dfrac{\tau_z}{I_z} \end{cases} \tag{3.18}$$

With this models, *LQR* controllers are computed for each subsystem. The obtained controller structure result to be 4 PD controllers in parallel. Where, following the same structure, the control laws are expressed as

i. *Pitch* control:

$$\tau_x = \begin{bmatrix} kP_\theta & kD_\theta \end{bmatrix} \begin{bmatrix} \theta_r - \theta \\ -\dot{\theta} \end{bmatrix} \tag{3.19}$$

where

$$\theta_r = \begin{bmatrix} kP_x & kD_x \end{bmatrix} \begin{bmatrix} x_r - x \\ \dot{x}_r - \dot{x} \end{bmatrix} \tag{3.20}$$

ii. *Roll* control:

$$\tau_y = \begin{bmatrix} kP_\phi & kD_\phi \end{bmatrix} \begin{bmatrix} \phi_r - \phi \\ -\dot{\phi} \end{bmatrix} \tag{3.21}$$

where

$$\phi_r = \begin{bmatrix} kP_y & kD_y \end{bmatrix} \begin{bmatrix} y_r - y \\ \dot{y}_r - \dot{y} \end{bmatrix} \tag{3.22}$$

iii. Altitude control:

$$T = \begin{bmatrix} kP_z & kD_z \end{bmatrix} \begin{bmatrix} z_r - z \\ \dot{z}_r - \dot{z} \end{bmatrix} + mg \tag{3.23}$$

iv. *Yaw* control:

$$\tau_z = \begin{bmatrix} kP_\psi & kD_\psi \end{bmatrix} \begin{bmatrix} \psi_r - \psi \\ \dot{\psi}_r - \dot{\psi} \end{bmatrix} \tag{3.24}$$

By combining equations (3.6) to (3.11) in a four parallel controllers, the control input $U = [T, \tau_x, \tau_y, \tau_z]$ is obtained. Let it be noted that for the the *pitch* and *roll* controllers an inner cascade structure appears due to the coupling of attitude and position variables.

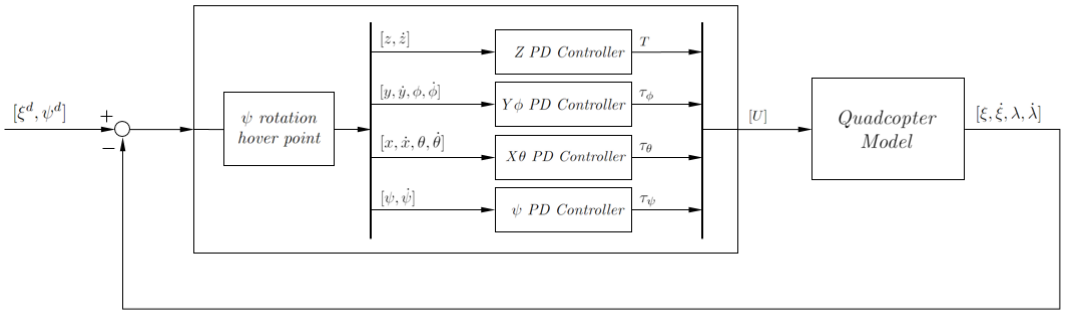


Figure 3.5: Model linearization control structure diagram.

In Figure 3.6 it can be seen that previous to computing the control input U a rotation around the ψ-axis is computed. This rotation must be considered as the controllers are derived from

linearizing around a hover point, ie. considering the xy-plane horizontal, which it does not imply that the xy-plane cannot rotate. equation (3.25) shows the rotation matrix applied to x and y-axis of the body frame.

$$R_{\psi-hover} = \begin{bmatrix} \cos\psi & \sin\psi \\ -\sin\psi & \cos\psi \end{bmatrix} \tag{3.25}$$

3.2 Backstepping controller

The non-linear control strategy explained in this section has been originally developed by R. Mahony and T. Hamel in [11].

The non-linear controller is based on a Lyapunov function derived from the backstepping technique [5, 7] and build up of the tracking position and attitude errors and its derivatives. Therefore, by applying the Lyapunov theorem the derived controller guarantees that all previously mentioned errors convergetowards zero, i.e. the system will converge in the tracking control problem.

In the mentioned paper, the control strategy designed is proposed for a scale model helicopter. The model approximates the dynamics by ignoring all small body forces and aerodynamic effects. Albeit these simplifications, the authors believe that resolving the basic trajectory plannings and control issues for the considered model, allows for extending the development to provide robust controllers for scale-model helicopter.

Even though the mentioned paper develops its research on a helicopter dynamic model, the derived controller can easily be implemented into a quadcopter as the inputs, in form of thrust and torques, are both the same for the dynamic models of both systems. By making use of equation (2.4) the model inputs are directly transformed into the quadcopter system inputs. Moreover, the simplifications done over the helicopter model are the same as the ones considered previously in Chapter 2.2 when modelling the dynamics of a quadcopter.

Next, the non-linear control structure will be derived by applying the backstepping technique. A control Lyapunov function will be built based on the pure feedback form [7] of the dynamic model of the quadcopter, equation (2.3), assuring the stability of the tracking control problem.

Problem

Find the control action $(T, \tau_x, \tau_y, \tau_z)$ dependant only on the measurable states $(\xi, \dot{\xi}, \lambda, \dot{\lambda})$ and arbitrarily many derivatives of the smooth trajectory $\left(\hat{\xi}(t), \hat{\psi}(t)\right)$ such that the tracking error

$$\mathscr{E} := \left(\xi(t)^d - \xi(t), \psi(t)^d - \psi\right) \in R^4$$

converges to zero for all possible initial conditions.

Deriving the controller

The recursive backstepping approach for the non linear controller starts by

- Defining the partial error

$$\delta_1 := \xi(t) - \hat{\xi}(t) \tag{3.26}$$

  which takes into account the tracking error relative to the position coordinates. The Lyapunov function will be composed by the sum of several storage function. Each storage

function will be designed as a Lyapunov function by itself. The first storage function is

$$S_1 = \frac{1}{2}\delta_1^T\delta_1 = \frac{1}{2}|\delta_1|^2$$

Taking the time derivative of S_1 and making use of the relation (2.3a)

$$\begin{aligned}\frac{d}{dt}S_1 &= \delta_1^T(\dot{\xi} - \dot{\hat{\xi}}) \\ &= \delta_1^T(v - \hat{v})\end{aligned}$$

where $\hat{v} := \dot{\hat{\xi}}$. Let v_d denote the desired value for velocity v so that

$$v_d := \hat{v} - \frac{1}{m}\delta_1 \tag{3.27}$$

With this choice the time derivative of the first storage function can be rewritten as

$$\begin{aligned}\dot{S}_1 &= \frac{1}{m}\delta_1^T(mv_d - m\hat{v}) + \frac{1}{m}\delta_1^T(mv - mv_d) \\ &= -\frac{1}{m}|\delta_1|^2 + \frac{1}{m}\delta_1^T(mv - mv_d)\end{aligned}$$

- The backstepping process continues by defining the new error

$$\delta_2 := mv - mv_d \tag{3.28}$$

where δ_2 also acts as a virtual control law for the previous backstepping step. The second storage function is

$$S_2 = \frac{1}{2}|\delta_2|^2 = \frac{1}{2}|mv - mv_d|^2$$

Deriving S_2 and making use of the relation (2.3a)

$$\begin{aligned}\dot{S}_2 &= \delta_2^T(m\dot{v} - m\dot{v}_d) \\ &= \delta_2^T(mge_3^A - R(\lambda)Te_3^B - m\dot{v}_d)\end{aligned}$$

At this point (λ, T) are the controlled variables of the backstepping controller. The input control variable T for the quadcopter model dynamics, equation (2.3), is explicit in the dynamics of S_2 and could be used for partially controlling the tracking error dynamics. Following this approach it leads to a time scale separation of the system dynamics [13] which is useful to ensure that one part of the system is more tightly controlled than others. This property is used in VTOL cases when hovering near to the ground. However, for more general trajectories, the derivative of the control variable T enter the remaining dynamics of the system and an aggressive control at this point could lead to extreme ill-conditioning of the remaining tracking control problem. Therefore, an alternative strategy is to consider the dynamics of T and include them in the control strategy

$$\ddot{T} = \tilde{T} \tag{3.29}$$

The control model input T and its first time derivative $\dot{T}$ become internal variables for the controller. this approach allows a better trade-off between the different control objectives for the tracking problem.

Let (λ_d, T_d) denote the desired values for (λ, T), and choose

$$\begin{aligned}X_d :&= R(\lambda_d)T_de_3^B \\ &= mge_3 - m\dot{v}_d + \delta_2 + \frac{1}{m}\delta_1\end{aligned} \tag{3.30}$$

The vectorial value of $R(\lambda_d)T_d e_3^B$ may be arbitrarily designed for any suitable values of (λ_d, T_d). With this relation, λ_d is not uniquely determined. The freedom that this is later used for choosing the control law of the *yaw* ψ. With the relation (3.30), the derivative of the second storage function can be rewritten as

$$\dot{S}_2 = -|\delta_2|^2 - \frac{1}{m}\delta_2^T\delta_1 + \delta_2^T(X_d - R(\lambda)Te_3^B)$$

- The backstepping process continues by defining the third error

$$\begin{aligned}\delta_3 :&= R(\lambda_d)T_d e_3^B - R(\lambda)Te_3^B \\ &= X_d - R(\lambda)Te_3^B\end{aligned}$$

as the vectorial difference between the desired and true values of the projected thrust that controls the position dynamics. An error for the *yaw* component is also defined

$$\varepsilon_3 = \psi - \hat{\psi} \tag{3.31}$$

The *yaw* component of the tracking control problem is introduced at this point of the control strategy, rather than at the start with the position error component δ_1, so that the relative degree between δ_3 and ε_3 would match with the control variables $\ddot{T}$ and τ.

The third storage function is defined as

$$S_3 = \frac{1}{2}|\delta_3|^2 + \frac{1}{2}|\epsilon_3|^2$$

Deriving S_3 and recalling equation 2.3b the following expression is obtained

$$\dot{S}_3 = \delta_3^T(\dot{X}_d - (R(\lambda)\dot{T}e_3^B + R(\lambda)Tsk(\Omega)e_3^B)) + \varepsilon_3(\dot{\psi} - \dot{\hat{\psi}}) \tag{3.32}$$

Note that the derivative of X_d is computed by differenciating the right term of the equation (3.30), while derivatives of T_d and λ_d are never explicitly computed.

– Considering the first term of the $\dot{S}_3$ expression. Let $(\dot{T}_d, \Omega_d)$ denote the desired values for the derivative $\dot{T}$ and Ω. Analogous to the vectorial assignation in (3.30), the vectorial term associated with δ_3 is assigned

$$\begin{aligned}Yd :&= R(\lambda)\dot{T}_d e_3^B + R(\lambda)Tsk(\Omega_d)e_3^B \\ &= \dot{X}_d + \delta_3 + \delta_2\end{aligned} \tag{3.33}$$

Recalling the anti-commutativity property of cross products

$$sk(\Omega_d)e_3 = \Omega_d \times e_3 = -sk(e_3)\Omega_d$$

the terms of equation (3.33) can be rewritten as

$$\begin{pmatrix} 0 & T & 0 \\ -T & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \Omega_d^1 \\ \Omega_d^2 \\ \dot{T}_d \end{pmatrix} = R(\lambda)^T(\dot{X}_d + \delta_3 + \delta_2) \tag{3.34}$$

The vector notation of the left term shows that any vector that results from the right part term will determine Ω_d^1, Ω_d^2 and $\dot{T}_d$. This leaves Ω_d^3 free to control independently the *yaw*.

Even thought it might seem that Ω_d^3 is independent of any virtual law, by recapping the kinematic relation between Euler angles and the angular velocity of the body,

$$\dot{\lambda} = \begin{pmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{pmatrix} \Omega = W(\lambda)^{-1}\Omega \tag{3.35}$$

where

$$W(\lambda) := \begin{pmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \sin\phi\cos\theta \end{pmatrix}$$

and replacing $\dot{\lambda}$ by $\dot{\lambda}_d$ and Ω by Ω_d, the expression obtained for $\dot{\phi}_d$ is

$$\dot{\phi}_d = \frac{\sin\phi}{\cos\theta}\Omega_d^2 + \frac{\cos\phi}{\cos\theta}\Omega_d^3$$

It is assumed that $\phi, \theta \in (-\pi/2, \pi/2)$. By reformulating the previous relation,

$$\Omega_d^3 = \frac{\cos\theta}{\cos\psi}\Big(\dot{\hat{\psi}} - \varepsilon_3 - \frac{\sin\phi}{\cos\theta}\Omega_d^2\Big)$$

the exact dependency of Ω_d^3 on Ω_d^2 can be seen. Therefore, due to the nonlinear couplings of the attitude dynamics, Ω_d^3 is partially dependant on equation (3.34).

– Now considering the second term of $\dot{S}_3$ expression associate with ε_3 where the input variable is the *yaw*. Let $\dot{\psi}_d$ denotes the desired *yaw* velocity and choose

$$\dot{\psi}_d := \dot{\hat{\psi}} - \varepsilon_3 \tag{3.36}$$

With all this choices made, the derivative of the third storage function can be rewritten as

$$\dot{S}_3 = -|\delta_3|^2 - \delta_3^T\delta_2 - \varepsilon_3^2 + \varepsilon_3(\dot{\psi} - \dot{\psi}_d) + \delta_3^T\big(Y_d - (R(\lambda)\dot{T}e_3^B + R(\lambda)Tsk(\Omega)e_3^B)\big)$$

- For the last backstepping step two new error are introduced

$$\delta_4 = Y_d - \big(R(\lambda)\dot{T}e_3^B + R(\lambda)Tsk(\Omega)e_3^B\big) \tag{3.37}$$

$$\varepsilon_4 = \dot{\psi} - \dot{\psi}_d \tag{3.38}$$

And the fourth storage function is defined by this new error:

$$S_4 = \frac{1}{4}|\delta_4|^2 + \frac{1}{2}|\varepsilon_4|^2$$

The derivative of S_4 is

$$\dot{S}_4 = \delta_4^T\Big(\dot{Y}_d - \big[r(\lambda)\ddot{T}e_3^B + 2R(\lambda)\dot{T}sk(\Omega)e_3^B + R(\lambda)T(\dot{\Omega}\times e_3^B)\big]\Big) + \varepsilon_4(\ddot{\psi} - \ddot{\hat{\psi}})$$

At this stage the control inputs of the model are accessible at the backstepping controller through: $\ddot{T} = \tilde{T}$, $\dot{\Omega}$ via equation (2.3b) and $\ddot{\psi}$.

For the purpose of simplifiction, a linearizing control input transformation is defined:

$$\tilde{\tau} := -J^{-1}\Omega \times J\Omega + J^{-1}\tau \tag{3.39}$$

With this transformation, equation (2.3b) is reduced to

$$\dot{\Omega} = \tilde{\tau} \tag{3.40}$$

If the second derivative of equation (3.35) is derived, the first components results in

$$\begin{aligned}\ddot{\psi} &= -e_1^T W(\lambda)^{-1}\dot{W}(\lambda)W(\lambda)^{-1}\Omega + -e_1^T W(\lambda)^{-1}\tilde{\tau} \\ &= -e_1^T W(\lambda)^{-1}\dot{W}(\lambda)W(\lambda)^{-1}\Omega + \frac{\sin\phi}{\cos\theta}\tilde{\tau}^2 + \frac{\cos\phi}{\cos\theta}\tilde{\tau}^3\end{aligned} \tag{3.41}$$

With equations (3.29) and (3.40) the derivative of the fourth storage function can be rewritten as

$$\dot{S}_4 = \delta_4^T\left(\dot{Y}_d - 2R(\lambda)\dot{T}sk(\Omega)e_3^B - \left[R(\lambda)\tilde{T}e_3^B - R(\lambda)T\tilde{\omega}e_3^B\right]\right) + \varepsilon_4(\ddot{\psi} - \ddot{\hat{\psi}})$$

where the desired control is obtained by choosing

$$R(\lambda)\tilde{T}e_3^B - R(\lambda)T\tilde{\omega}e_3^B = \dot{Y}_d - 2R(\lambda)\dot{T}sk(\Omega)e_3^B + \delta_3 + \delta_4 \tag{3.42}$$

$$\ddot{\psi} = \ddot{\hat{\psi}} - \varepsilon_4 - \varepsilon_3 \tag{3.43}$$

where both equations have to be satisfied at the same time. By rewritten equation (3.42) analogous to equation (3.35):

$$\begin{pmatrix} 0 & T & 0 \\ -T & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix}\tilde{\tau}^1 \\ \tilde{\tau}^2 \\ \tilde{T}_d\end{pmatrix} = R(\lambda)^T\left(\dot{Y}_d - 2R(\lambda)\dot{T}sk(\Omega)e_3^B + \delta_3 + \delta_4\right) \tag{3.44}$$

and as long as $T \neq 0$, the control signals $\tilde{\tau}^1$, $\tilde{\tau}^2$ and $\tilde{T}$ and uniquely determined. By solving equation (3.41), in the form of

$$\frac{\cos\phi}{\cos\theta}\tilde{\tau}^3 = \ddot{\hat{\psi}} - \varepsilon_4 - \varepsilon_3 + e_1^T W(\lambda)^{-1}\dot{W}(\lambda)W(\lambda)^{-1}\Omega - \frac{\sin\phi}{\cos\theta}\tilde{\tau}^2 \tag{3.45}$$

$\tilde{\tau}^3$ is uniquely determined for a known $\tilde{\tau}^2$.

The above process uniquely determines $\tilde{T}$, $\tilde{\tau}^1$, $\tilde{\tau}^2$ and $\tilde{\tau}^3$. Using equations (3.29) and (3.39) the original control inputs T and τ can be recovered.

The backsteping procedure allows to build the Lyapunov function

$$\mathscr{L} := S_1 + S_2 + S_3 + S_4$$

where it derivative can easily be computed as

$$\begin{aligned}\dot{\mathscr{L}} &= \dot{S}_1 + \dot{S}_2 + \dot{S}_3 + \dot{S}_4 \\ &= -\frac{1}{m}|\delta_1|^2 - |\delta_2|^2 - |\delta_3|^2 - |\varepsilon_3|^2 - |\delta_4|^2 - |\varepsilon_4|^2\end{aligned}$$

By direct inspection it can be seen that the Lyapunov function is always decreasing.

As a recall, δ_1 and ε_3 form the original tracking error. The error δ_2 regulates the linear velocity, the additional errors δ_3 and δ_4 add information about the *pitch*, *roll* and their derivatives, while ε_3 and ε_4 regulate the *yaw* and its derivative independently.

To synthesise the controller developed above, and firstly assume variables δ_1, δ_2, X_d, δ_3, Y_d, δ_4, ε_3 and ε_4, along with the times derivative of some of these, as already defined up to this point,

the control laws for the control inputs can be derived from the virtual control laws defined in equations (3.44) and (3.45) and shown below

$$\begin{pmatrix} 0 & T & 0 \\ -T & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \tilde{\tau}^1 \\ \tilde{\tau}^2 \\ \tilde{T} \end{pmatrix} = R(\lambda)^T \left(\dot{Y}_d - 2R(\lambda)\dot{T}sk(\Omega)e_3^B + \delta_3 + \delta_4 \right)$$

$$\frac{\cos\phi}{\cos\theta}\tilde{\tau}^3 = \ddot{\hat{\psi}} - \varepsilon_4 - \varepsilon_3 + e_1^T W(\lambda)^{-1}\dot{W}(\lambda)W(\lambda)^{-1}\Omega - \frac{\sin\phi}{\cos\theta}\tilde{\tau}^2$$

From these equations the virtual control variables $\tilde{T}$, $\tilde{\tau}^1$, $\tilde{\tau}^2$, $\tilde{\tau}^3$ are obtained. By recovering equation (3.29) and (3.39):

$$\ddot{T} = \tilde{T}$$

$$\tilde{\tau} := -J^{-1}\Omega \times J\Omega + J^{-1}\tau$$

the control variables can easily be computed:

$$T = \iint \tilde{T} dt$$

$$\begin{pmatrix} \tau_x \\ \tau_y \\ \tau_z \end{pmatrix} = J \begin{pmatrix} \tilde{\tau}_1 \\ \tilde{\tau}_2 \\ \tilde{\tau}_3 \end{pmatrix} + \Omega \times J\Omega$$

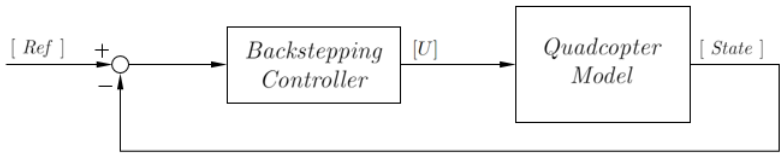


Figure 3.6: Non-linear control structure diagram.

Chapter 4

Simulations

In this chapter the dynamics of the quadcopter will be simulated for a tracking control problem. The simulations will be carried out with the different control structures derived in Chapter 3. Previous to the simulations some information about the drone Parrot AR Drone 2.0 is mentioned, as the simulations are meant to be done for this type of drone. Later, some graphs of the response of the system for each simulation will be shown. For each controller, the simulations will be run twice: first with the reference for the tracking variables and second with the reference of the tracking variables and its first derivative.

4.1 Preliminaries

The simulations carried out in this chapter aim to simulate the response of the drone Parrot AR Drone 2.0 [16] for the tracking problem studied. Therefore, some parameters must be known in order to perform these simulations. In the table 4.1 the mechanical parameters of the quadcopter are shown.

For an accurate simulation of the quadcopter response, the actuator saturation should be considered. The desired thrust T and torques τ must be constrained by the maximum feasible motor speeds by the relation (2.4). The constants c_T and c_Q involved in the previous relation are found experimentally [14, 20].

The maximum rotor speed is approximately 470 rad/s producing a thrust of 1.8 N per rotor.

Table 4.1: Mechanical parameters

Parameters	Value	Unit
g	9.807	$m \cdot s^{-2}$
m	0.429	kg
I_x	$2.237 \cdot 10^{-3}$	$kg \cdot m^2$
I_y	$2.985 \cdot 10^{-3}$	$kg \cdot m^2$
I_z	$4.803 \cdot 10^{-3}$	kg
l	0.1785	m
c_T	$8.048 \cdot 10^{-6}$	
c_Q	$2.423 \cdot 10^{-6}$	

The saturation function results in

$$T = \begin{cases} T & \text{if } T < 7.2 \\ 7.2 & \text{if } T \geq 7.2 \end{cases} \tag{4.1}$$

When the quadcopter hovers, thrust is $T = mg$ and the torques $\tau_i = 0$, which equals to a speed of approximately 360 rad/s at all rotors.

The absolute torque limit, depends on the T and the quadcopter geometry. For Parrot AR Drone 2.0 is approximately 0.1397 Nm for τ_x and τ_y and 0.028 Nm for τ_z.

The designed controllers are meant to be integrated in the quadcopter. Therefore, the simulations are computed considering that the controller will be discrete and the states will be sampled when measured for feedback. Due to the different sensors included in the hardware of the Parrot AR Drone 2.0 [16, 9] the state measurement are done at different frequencies. The position and *yaw* states are obtain 25 Hz while the *pitch* and *roll* are obtained at 200 Hz.

4.2 Trajectory generation

As mentioned previously, the tracking control problem relates to the position and *yaw* states. For the simulations carried out several trajectories are generated along these states. For the *pitch* and *roll* states no trajectory is generated. However, as it will be shown, for some control structures the desired states are computed internally at the controller.

In order to generate a trajectory from an initial to a final point an important characteristic of the quadcopter dynamics is taken into account: motor commands are proportional to the attitude accelerations, or forth derivative of the path [4, 2]. By generating the trajectories from the continuous forth derivative of itself a smooth trajectory is obtained, Figure (4.1). Moreover, knowing and controlling the forth derivative of the trajectory allows to have a quantitative notion of the expected trajectory, the time in which it can be accomplished, and avoids discontinuities and abrupt changes in the control inputs.

If the trajectory is desired to be time-optimal several approaches have been derived [8]. However, still there is no single algorithm, capable of planning and optimising a quadcopter path along a domain with obstacles.

For the computed trajectories the initial and final point are assumed to be steady state. And because the way they are computed, the derivatives of the trajectory are also available as a reference.

For all cases shown in this chapter the trajectory generated for the tracking problem is for variables (x, y, z, ψ) goes from point $(0, 0, 0, 0)$ to point $(1, 2, -4, \pi/3)$.

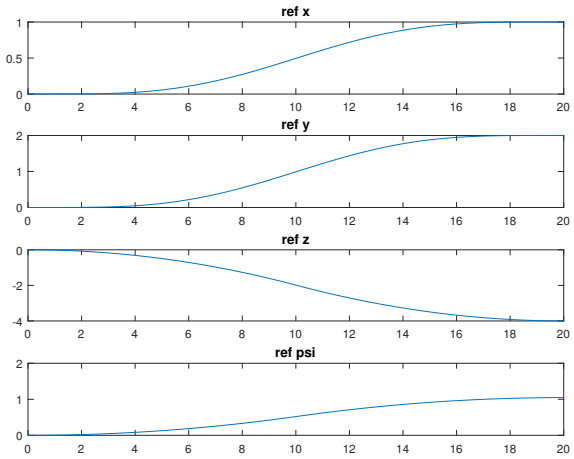


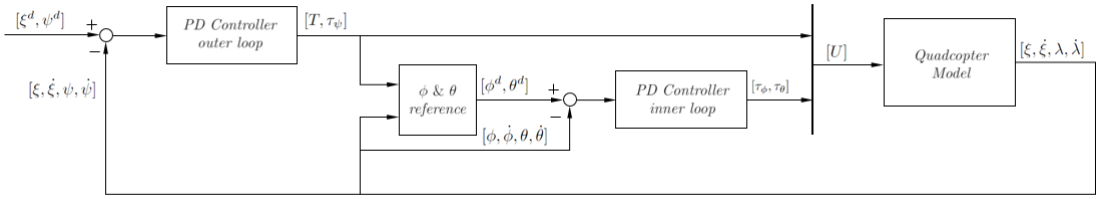
Figure 4.1: Smooth reference trajectory generated

4.3 Simulations

The simulations are computed in the Simulink enviroment *Matlab*. The numerical solver used is the one by default ode45 with variable step.

4.3.1 PD - cascade control approach

The first controller simulated is the cascade controller from section 3.1 with a control structure of Figure (3.4).



The controller constants for this cascade controller are found at table 4.2. For this controller strategy, the inner loop runs at 200 Hz and the outer loop at 25 Hz. Due to the cascade structure, as mentioned previously, the desired ϕ^d and θ^d are computed as they are needed explicitly as inputs for the inner loop controller.

Several simulations have been run with this controller. As this controller is linear, the first

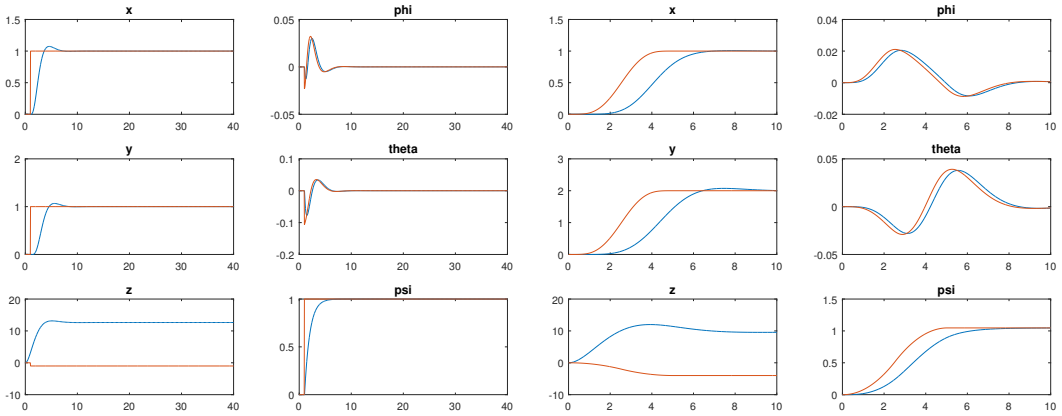
Table 4.2: Constants for cascade controller

	K_P	K_D
K_x	0.3085	0.5237
K_y	0.3085	0.5237
K_z	-0.3085	-0.5237
K_ψ	0.1044	0.1091
K_ϕ	0.3958	0.0980
K_θ	0.4070	0.1035

simulation will show the response of the the controller with the simplified model (3.2) in order to prove that the controller does control the system it was derived from. The second simulation will implement the same control structure but with the non-linear dynamics of the quadcopter model, which is the system that we really aim to control. As the trajectories to be tracked are continuous up to its fourth derivative, and these derivatives are available, a third simulation will be done including the first derivative of the trajectory as a reference to see if there is a significant improvement.

Response of the simplified system

As a first approach to the derived controller some simulations have been carried out with the simplified model, equation (3.2), it was derived from.



(a) State response to a step input (b) State response to a smooth trajectory

Figure 4.2: Response of the simplified model with a cascade control

In Figure (4.2a) the response of the simplified system to a unitary step input can be seen. The system converges, except for the z-axis where a constant error appears, and its stabilised at a hovering point. It must be noted that in z-axis and ψ-axis the system converges without any oscillation, while in the x-axis and y-axis the dynamics do show small oscillations.

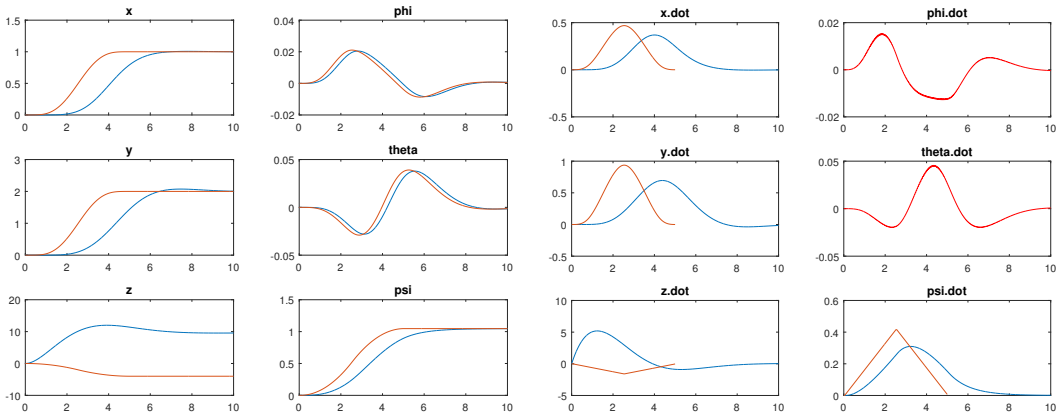
In Figure (4.2b) the response of the system dynamics to a given path is shown. The states of the system follow the given trajectory with a small delay. The delay of the attitude response is smaller than the position response showing that the inner loop converges faster.

It can be said that the linear controller is properly designed and implemented and can deal with a regulation and a tracking control problem. However, the controller is aimed to control the non-linear model of the quadcopter.

Trajectory tracking with full-order dynamics

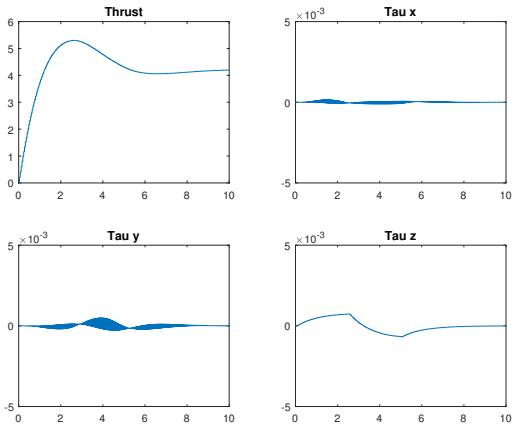
By implementing the same control structure as in previous section but with the full-order dynamics of the model, equation (2.3), the results obtained are shown in Figure (4.3).

It can be seen how the system variables follow the given trajectory. Even thought the full-order dynamic model has been implemented the results obtained of the trajectory tracking do not differ significantly from the simplified model results. Moreover, the same problem can be observed as the response presents a constant error in z-axis due to initial oscillation of the thrust.



(a) States dynamics response

(b) States derivatives response



(c) Control actions dynamics

Figure 4.3: Response of the full-order model with a cascade control

Trajectory tracking with full-order dynamics to a trajectory and its derivative

Introducing the derivative of the pre-computed trajectory as a reference to track, the dynamics of the system are shown in Figure (4.4).

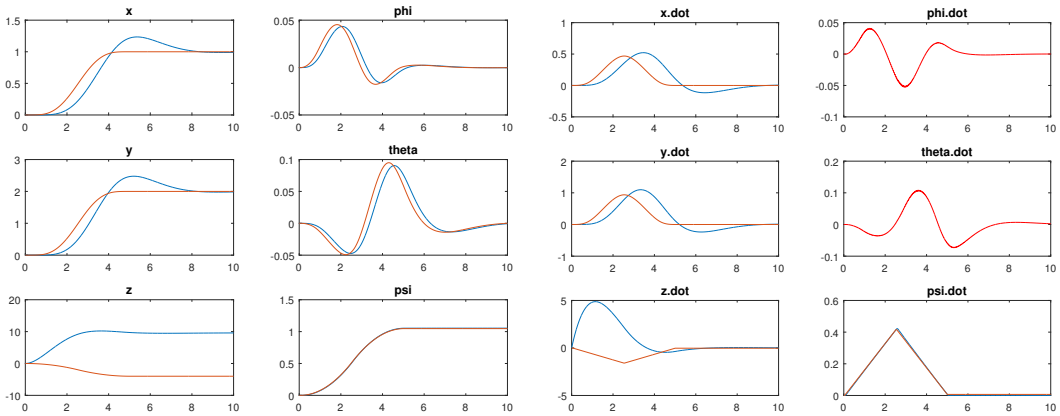
With respect to Figures (4.3) of the previous section, Figure (4.4) shows that including the derivative of the trajectory translates into a better response of the system dynamics, a smaller delay between the trajectory and the system variables and faster control actions. Even though there is an improvement, the z-axis continuous to present a constant error.

It can be concluded that, due to the error in the z-axis, the control structure is not suitable. However, if this error is cancelled, the response of the other variables indicate this approach could be suitable.

4.3.2 PD - Model linearization approach

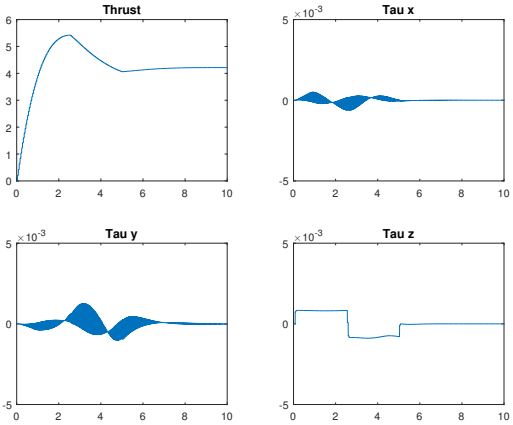
The second controller to be simulated is the 4 PD controllers in parallel from section 3.1.1 with a control structure like in Figure (3.6).

The controller constants for these 4 PD controllers in parallel are shown in table 4.3.



(a) States dynamics response

(b) States derivatives response



(c) Control actions dynamics

Figure 4.4: Response of the full-order model with a cascade control and including the derivative of the trajectory as reference

For this constants the loop is supposed to run a 25 Hz even though it is not exactly true as some state variables are sampled at a higher ratio. However, as the subsystem controllers have coupled variables, all are supposed to run at the lowest frequency. This situation could be handled with a state estimator or a zero-order-holder.

Due to the structure of the controller, the desired ϕ^d and θ^d do not have to be computed explicitly. The control structure has one feedback loop where the references correspond to the tracking path. The coupled dynamics will lead to the necessary values of ϕ and θ so that the trajectory can be tracked.

As in the previous section 4.3.1, several simulations have been carried out. As this controller is also linear, the first simulation will show the response of the the controller with the linearised model of the quadcopter, equation (3.14), to prove that the controller is correctly designed and implemented. A second simulation will implement the same controller with the non-linear dynamics of the quadcopter model. After, a third simulation is done to see how including the derivative of the trajectory improves the systems dynamics response.

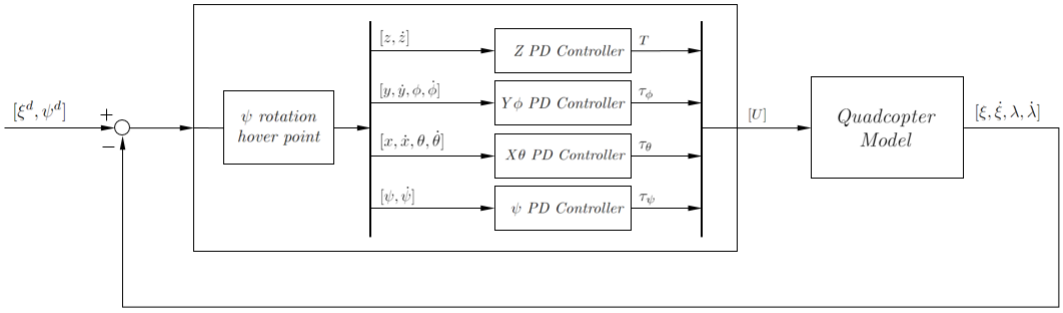
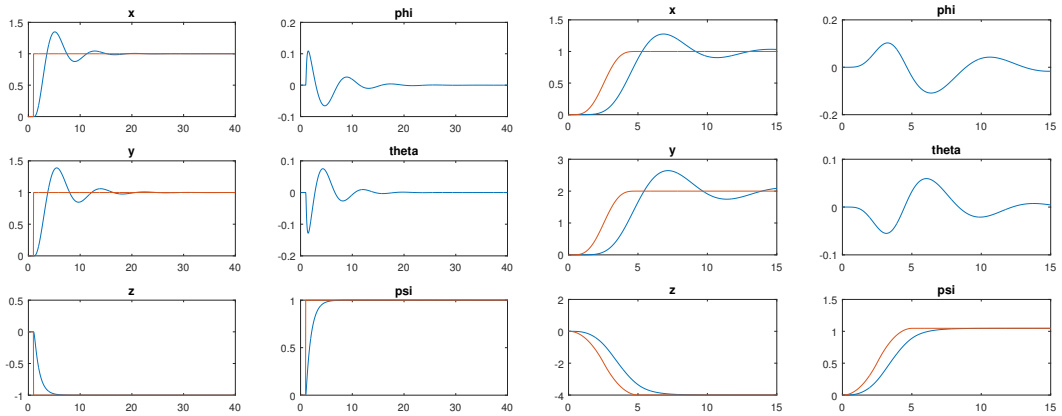


Table 4.3: Constants for linearized model controller

	K_P	K_D
K_x	-0.1526	-0.1501
K_θ	0.2787	0.0627
K_y	0.1279	0.1251
K_ϕ	0.2307	0.0513
K_z	-2.6794	-3.0787
K_ψ	0.0654	0.0700

Response of the linearized model

As a first approach to the derived controller some simulations have been carried out with the linearised model, equation (3.14), it was derived from.



(a) State response to a step input

(b) State response to a smooth trajectory

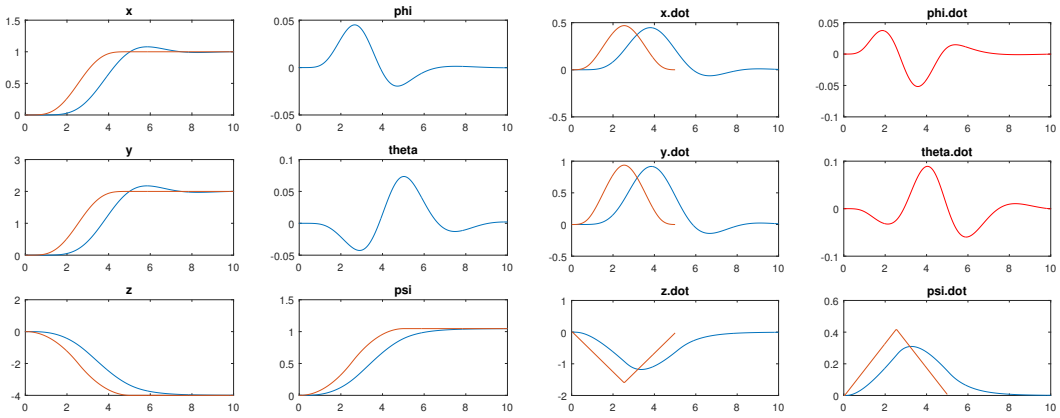
Figure 4.5: Response of the linearised model with a linearized model controller

In Figure (4.5a) the response of the simplified system to a unitary step input can be seen. It can be seen that all variables converge and stabilize at the hovering point given. As with the previous PD controller, in z and ψ axis the system converges without oscillating, while in x and y axis it does present oscillations.

In Figure (4.5b) the response of the system dynamics to a given path is shown. The states of the system follow the given trajectory with a small delay. Again, no reference is given for ϕ and θ states.

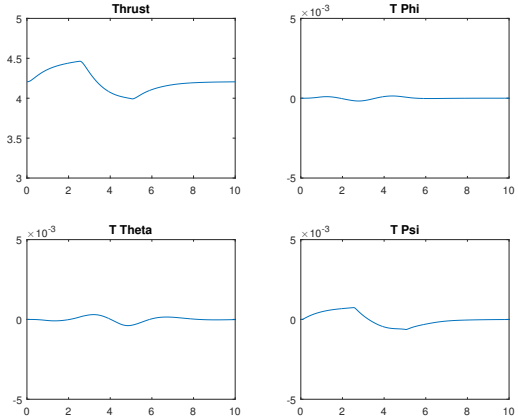
Trajectory tracking with full-order dynamics

By implementing the same control structure as in previous section but with the full-order dynamics of the model, equation (2.3), the results obtained are shown in Figure (4.6).



(a) States dynamics response

(b) States derivatives response



(c) Control actions dynamics

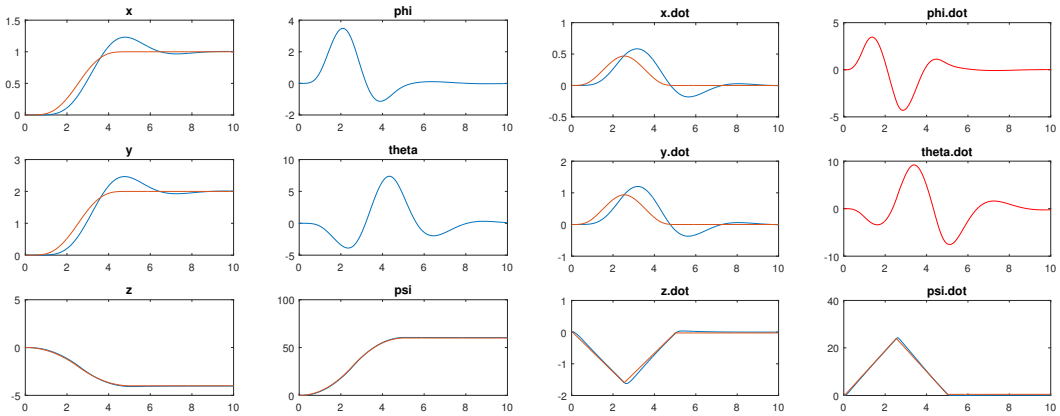
Figure 4.6: Response of the full-order model with a model linearization control

It can be seen how the system variables follow the given trajectory. Moreover, the state derivatives show a more sharp response, reasonable with the non-linear dynamics of the system. Also, the control actions present a more non-linear response, especially the control actions responsible for the *pitch* and *roll*.

Trajectory tracking with full-order dynamics to a trajectory and its derivative

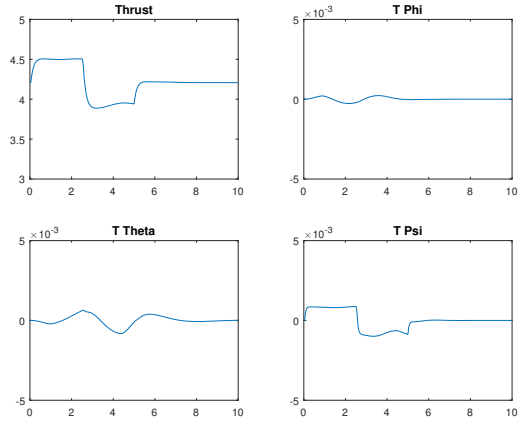
Introducing the derivative of the pre-computed trajectory as a reference to track, the dynamics of the system are the ones shown in Figure (4.7).

With respect to Figures (4.6) of the previous subsection, Figure (4.7) shows that including the derivative of the trajectory translates into a better response of the system dynamics, a smaller delay between the trajectory and the system variables and faster control actions.



(a) States dynamics response

(b) States derivatives response



(c) Control actions dynamics

Figure 4.7: Response of the full-order model with a model linearization control and including the derivative of the trajectory as reference

4.3.3 Backstepping controller

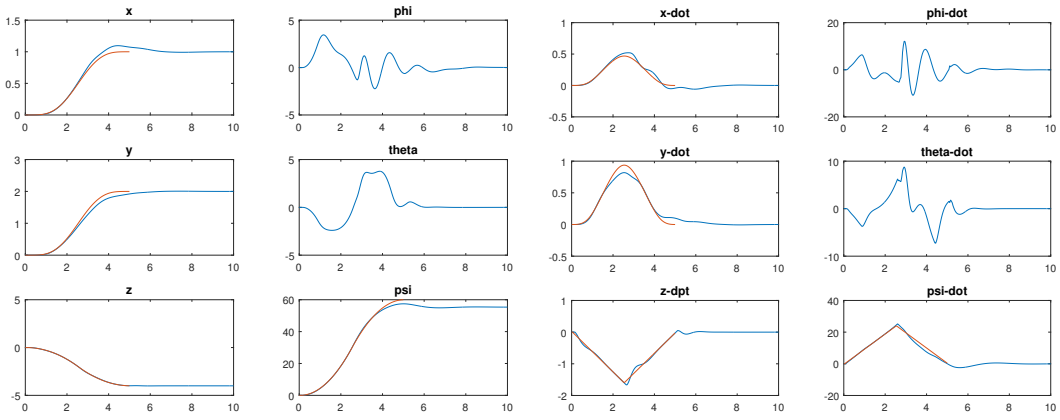
The third control structure to be simulated is the non-linear controller derived in section 3.2. This controller is the result of applying the backstepping process in order to control the quadrotor model. A Lyapunov function that includes the tracking errors ensures that adequate response of the controller.

Also, the structure of the controller does not need the desired ϕ^d and θ^d to be computed explicitly. However, it does need the second derivative of the ψ as a reference.

As in the previous controller, the feedback loop is run at 25 Hz due to the same reasons.

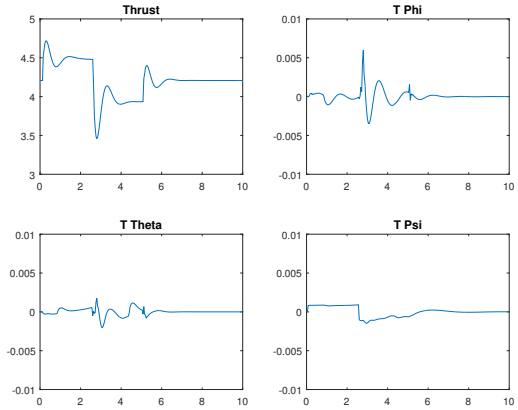
For this case, as it has been mentioned, the control structure includes the non-linearities of the model and requires as reference the trajectory and some of its derivatives, therefore the controller is simulated directly with the full-order model.

The response of the system is shown in Figure (4.8). It can be seen that the response of the system, a seen in subfigure (4.8a), is very fast and has nearly no delay with respect to the reference trajectory. Furthermore, as it could be expected, the response of the system does show the non-linearities included in the control structure: the control actions presents a fast and highly non-linear response. Moreover, the derivarives of ϕ and θ also present some small sharp edges due



(a) States dynamics response

(b) States derivatives response



(c) Control actions dynamics

Figure 4.8: Response of the full-order model with the backstepping controller

to this non-lineatities and the fact that no reference is generated for them.

In can be concluded that the best control structure derived up to this point is the no-linear controller derived in the last section, as it presents the best response of the system. However, it must be said that the response of the system, modelled with high non-lineraties, does res have a good performance when it is controlled with a linear controller like in section 4.3.2.

Chapter 5

Conclusions

In this thesis several topics pertaining to the control of a quadcopter have been tackled. In the previous chapters a model of a quadcopter was derived and controller using different control structure approaches as it has been shown in the different simulations.

In Chapter 2 a nonlinear model of the quadcopter was obtained from the Newton-Euler formulation. The aerodynamics effects, as well as small forces in the body frame were not considered when describing the dynamics of the position and attitude of the quadcopter. The coupling of the state variables was mentioned, as well as the implications it had for deriving different control structures. In Chapter 3 different controllers, linear and nonlinear, have been formulated as a result of the different approaches to the tracking problem and the model obtained in Chapter 2. In Chapter 4 several simulation had been carried out for the different control strategies showing how the response of the system varied depending on the control strategy followed.

5.1 Future work

Some recommendation are made in order to set a path for the continuing of this project.

- PD - Cascade strategy approach

  It would be interesting to see how the constant error in the z-axis can be eliminated by the controller.

- Simulation

  The simulations could be improved and made more realistic introducing noise to the measurements. Also, more limit and abrupt manoeuvres could be track in order to test the robustness of the derived controllers.For a better handling of limit cases anti-windup loops should be include to assure a faster convergence and stability.

- Nonlinear controllers

  While the classic linear approaches have been extensively studied and derived, the field of nonlinear control is much more wide and unexplored. Therefore the implementation of different nonlinear control structures is highly encouraged.

– State estimator/observer

  The design of a state estimator would not only take the simulation towards a more realistic scenario of implementation but would help dealing with the noise in the measurements and could also help dealing with the different sampling rates.

– Implementation

  For studying the response of a quadcopter and verifying all the simulations the implementation of the control strategies in a Parrot AR Drone 2.0 as well as to calibrate the accurateness of the model is believed to be crucial.

– Auto calibration for experimental constants

  As a precise automated system, the quadcopter experimental constants, such as c_T and c_Q, have to be calibrated. Due to the research phase in which the quadcopter will be controlled, failures and crashes will appear to be inevitable. Therefore an automated auto calibration is believed to be an interesting and useful problem to deal with.

Bibliography

[1] L. Villani G. Oriolo B. Siciliano, L. Sciavicco. *Robotics Modelling, Planning and Control*. 2008. 1

[2] N. Roy C. Richter, A. Bry. Polynomial Trajectory Planning for Quadrotor Flight. 22

[3] P. Corke. *Robotics, Vision and Control*. 2011. 1

[4] Vijay Kumar Daniel Mellinger. Minimum Snap Trajectory Generation and Control for Quadrotors. 22

[5] Ola Harkegard. Flight Control Design Using Backstepping. Master's thesis, Linkopings universitet, 2001. 15

[6] A.S. Morse I. Kanellakopoulos, P.V. Kokotovic. *Systematic design of adaptative controllers for feedback linearizable systems*. 1990. 5, 7

[7] M. Krstić I. Kanellakopoulos, P.V. Kokotovic. *Nonlinear and adaptative control design*. 1995. 15

[8] P. Lambrechts. Trajectory planning and feedforward design for electromechanical motion systems. 2003. 22

[9] D. Vissière N. Petit P. Bristeau, F. Callou. The Navigation and Control technology inside the AR Drone micro UAV. 2011. 22

[10] R. Lozano J. Ostrowski R. Mahony, T. Hamel. Dynamic modelling and configuration stabilization for an X4-flyer. 2002. 5

[11] T. Hamel R. Mahony. Robust trajectory tracking for a scale model autonomous helicopter. 2004. 15

[12] T. Hamel R. Mahony, Sung-Han Cha. A coupled estimation and control analysis for attitude stabilisation of mini aerial vehicles. 2006. 5

[13] P.V. Kokotović R. sepulchre, M. Janković. *Constructive Nonlinear Control*. 1997. 16

[14] P.L.M Rooijakkers. System identification and control of a quadrotor. Master's thesis, Eindhoven University of Technology, 2016. 6, 21

[15] Roland Siegwart Samir Bouabdallah. Backstepping and Sliding-mode Techniques Applied to an Indoor Micro Quadrotor. 2005. 5

[16] Pierre Eline Frederic DHaeyer Stephane Piskorski, Nicolas Brulez. AR Drone Developer Guide. 2015. 21, 22

[17] the Engineer. textttthttps://www.theengineer.co.uk/flying-taxi-tests-to-begin-this-year-claims-airbus-boss/. January 2017. 1

[18] the Engineer. texttthttps://www.theengineer.co.uk/amazon-delivery-drones-get-green-light-for-uk-trials/. July 2016. 1

[19] the Engineer. texttthttps://www.theengineer.co.uk/issues/october-2012-online/roundtable-development-of-civilian-uavs/. October 2012. 1

[20] B.C.M. van Aert. Control and coordination algorithms for autonomous multi-agent quadrotor systems. Master's thesis, Eindhoven University of Technology, 2016. 21

[21] Holger Voos. Nonlinear Control of a Quadrotor Micro-UAV using Feedback Linearization. 5

[22] Holger Voos. Nonlinear Stat-Dependent Riccatu Equation Control of Quadcopter UAV. 5